

Федеральное государственное образовательное бюджетное учреждение
высшего профессионального образования
«Нижегородский Государственный Университет им.
Н.И.Лобачевского» (ННГУ)
Институт Информационных Технологий Математики и Механики

Отчёт по лабораторной работе
Работа с массивами случайно сгенерированных чисел

Выполнил:
студент группы 3821Б1ФИЗ

Мурадов К.Р.

Проверил:
заведующий лабораторией суперкомпьютерных
технологий и высокопроизводительных
вычислений

Лебедев И.Г

Нижний Новгород
2021г.

Содержание

Введение	1
Постановка задачи	1

Руководство пользователя	2
Руководство программиста	2
Описание структуры кода программы	2
Описание структуры данных	3
Описание алгоритмов	4
Эксперименты	Error! Bookmark not defined.
Заключение	6
Список литературы	6
Приложение 1	7

Введение

Программирование — это интересный, полезный и увлекательный процесс, благодаря которому создаются программы – набор инструкций, которые приводятся в исполнение компьютерами. Одной из ключевых задач компьютера является работа с данными. В том числе и со случайно генерируемыми наборами данных, о которых пойдет речь в настоящей работе.

Случайные числа — это одна из основных составляющих любого языка программирования, на них строятся многие алгоритмы. Они имеют применение в физике, например, в исследованиях электронного шума, в инженерном деле и исследовании операций. Многие методы статистического анализа требуют использования случайных чисел.

В ходе выполнения лабораторной работы на языке программирования «С» будет написана программа, работающая со случайными числами.

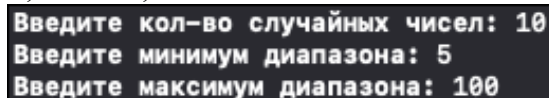
Постановка задачи

Программа генерирует множество случайных чисел размера n в диапазоне (\min , \max), где n , \min , \max вводятся с клавиатуры. После чего подсчитывает, выводит сумму, которая получается следующим образом: все числа, номера которых совпадают с дробной частью одного из исходных чисел - вычитаются, все остальные прибавляются.

Руководство пользователя

После запуска программа выводит три сообщения «Set N, min, max» означающим, что от пользователя для дальнейшей работы требуется ввести: число элементов массива случайных чисел в количестве n, нижнюю границу генерации чисел min и верхнюю max, в одну строку через пробелы, как указано в сообщении от программы (см. рис. 1).

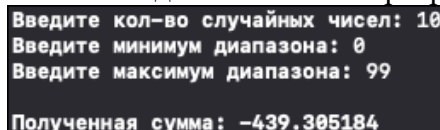
Например, введем $n = 10$, $\min = 5$, $\max = 100$



```
Введите кол-во случайных чисел: 10
Введите минимум диапазона: 5
Введите максимум диапазона: 100
```

Рисунок 1. Терминал после запуска

После нажатия Enter, программа выведет результат суммирования сгенерированных чисел, произведенного по правилам технического задания. На этом программа завершается (см. рис. 2).



```
Введите кол-во случайных чисел: 10
Введите минимум диапазона: 0
Введите максимум диапазона: 99
Полученная сумма: -439.305184
```

Рисунок 2. Результат работы программы

Руководство программиста

Описание структуры кода программы

1. Подключение библиотек, с которыми предстоит работать.

```
#include <stdio.h>
#include <stdlib.h>
```

2. Объявление функции main(), получение данных от пользователя.

```
int main(){

    int n = 0, min = 0, max = 0, i = 0, indexOfNumber = 1;
    double sum = 0.0;
    double* randomNumber;
    int* randomNumberDecimal;

    printf("Введите кол-во случайных чисел: ");
    scanf("%d", &n);

    printf("Введите минимум диапазона: ");
    scanf("%d", &min);

    printf("Введите максимум диапазона: ");
    scanf("%d", &max);

    printf("\n");
```

3. Создание динамического массива размера n типа float randomNumber и заполнение его случайно сгенерированными в заданном пользователем диапазоне. А также создание динамического массива randomNumberDecimal, который будет заполнен дробными частями случайных чисел.

```
randomNumber = (double*)malloc(n * sizeof(double));
randomNumberDecimal = (int*)malloc(n * sizeof(int));

// создаем randomное число и узнаем десятичную часть числа
for (i = 0; i < n; i++) {

    randomNumber[i] = RandomFloat(min, max);
    randomNumberDecimal[i] = (randomNumber[i] -
(int)randomNumber[i]) * 1000000;

}
```

4. Функция генерирующая случайно десятичное число в количестве n, с диапазонами (min, max)

```
double RandomFloat(int min, int max){
return ((max - min) * ((double)rand() / RAND_MAX)) + min;
}
```

5. Вывод суммы и проверка условия суммирования которые работают по следующему принципу:
Если десятичное число совпадает с индексом числа, то они вычитаются. Все остальное складывается.

```
// проверяем условия для суммирования
for (i = 0; i < n; i++){

    if (randomNumberDecimal[i] == indexOfNumber) {
        sum -= randomNumber[i];
    }
    else {
        sum += randomNumber[i];
    }

    indexOfNumber++;
}

// вывод суммы
printf("Полученная сумма: %f\n", sum);
return 0; // программа успешно завершена
```

Описание структуры данных

- В программе используются переменные счетчики типа int(i), количество генерируемых чисел n типа int, а также функция RandomFloat(), генерирующая случайное число типа double.
- Переменные min, max относятся к типу данных int, а sum относится к типу данных float
- Случайные числа, создаваемые в ходе работы программы, записываются в массивы типа double randomNumber, а дробные части этих чисел записываются в массив типа int randomNumberDecimal.

- Для функционирования программы требуются библиотеки: `stdio.h` и `stdlib.h`.

Описание алгоритмов

В данной программе реализовано несколько ключевых алгоритмов:

- Алгоритм заполнения массива случайными числами представлен следующей блок-схемой:

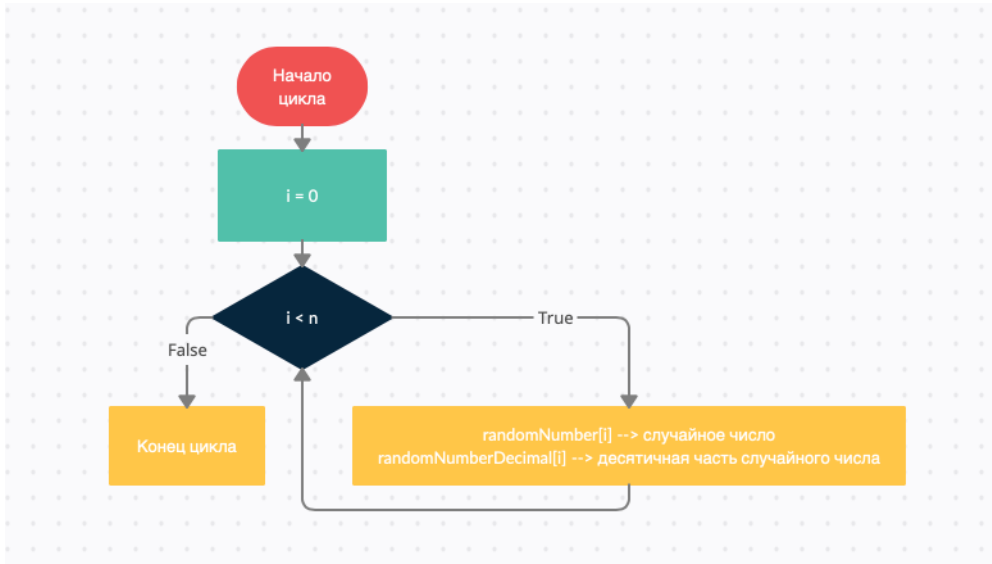


Рисунок 3. Блок-схема заполнения массива случайными числами

Алгоритм суммирования. Функционирует цикл, который будет работать n раз, проверяется дробное число с индексом числа. На данном этапе производятся действия: Если десятичное число равно с индексом числа, то число вычитается из суммы, иначе суммируется.

```

// проверяем условия для суммирования
for (i = 0; i < n; i++){

    if (randomNumberDecimal[i] == index0fNumber) {
        sum -= randomNumber[i];
    } else {
        sum += randomNumber[i];
        index0fNumber++;
    }

    // вывод суммы
    printf("Полученная сумма: %f\n", sum);
    return 0; // программа успешно завершена
  
```

Эксперименты

Написанную программу протестируем на различных входных данных, чтобы убедиться в её работоспособности в различных условиях.

1. Работа программы в случае ввода данных (см. рис. 4).

```
Введите кол-во случайных чисел: 100
Введите минимум диапазона: -1
Введите максимум диапазона: 9
Полученная сумма: 418.424689
```

Рисунок 4. Результат работы программы при входных данных

2. На рисунке 6 представлен более развернутый вариант работы программы, при котором в терминале выводятся сгенерированные числа. На этом примере мы можем наглядно убедиться в том, что при выполнении программы был сгенерирован набор случайных чисел, а затем была подсчитана их сумма, в соответствии с правилами поставленными в задаче (см. рис. 5).

```
Введите кол-во случайных чисел: 10
Введите минимум диапазона: 1
Введите максимум диапазона: 10
1.000070
2.183840
7.800448
5.127851
5.794905
2.970633
1.423402
7.109782
7.113667
9.412236
Полученная сумма: 49.936835
```

Рисунок 5. Развернутый вариант работы

3. Теперь попробуем подать на вход различные корректные значения (см. рис. 6-8).

```
Введите кол-во случайных чисел: 1000
Введите минимум диапазона: -9999
Введите максимум диапазона: 9999
Полученная сумма: -40768.339279
```

Рисунок 7. Работа программы с отрицательными числами

```
Введите кол-во случайных чисел: 5000000
Введите минимум диапазона: 1
Введите максимум диапазона: 99
Полученная сумма: 250086555.265374
```

Рисунок 8. Работа программы при больших N

```
Введите кол-во случайных чисел: 50000000
Введите минимум диапазона: -99999
Введите максимум диапазона: 99999
Полученная сумма: 164950844.675819
```

Рисунок 9. Работа программы при больших входных данных

При тестировании программы с различными входными данными в среде разработки Xcode никаких ошибок или предупреждений встречено не было. В среднем программа даже для больших входных данных выполнялась за небольшое время, не превышающее двух секунд для крайне больших значений n.

Заключение

В ходе лабораторной работы мною была написана программа на языке «С», которая генерирует множество случайных чисел размера n в диапазоне (\min , \max), где n , \min , \max вводятся с клавиатуры. После чего подсчитывает, выводит сумму, которая получается следующим образом: все числа, номера которых совпадают с дробной частью одного из исходных чисел - вычитаются, все остальные прибавляются.

Выполнение данной лабораторной работы позволило мне лучше изучить принципы работы с динамическими массивами, изучить различные алгоритмы позволяющие работать с данными, содержащимися в массивах, принципы работы со случайными числами, принципы работы с вводимыми пользователем данными их реализации внутри программы, их интерпретацию и проверку на корректность.

Выполнение подобных работ позволяет значительно улучшить свои навыки программирования, а также улучшить понимание работы с компьютером и средой разработки, в тоже время позволяет закрепить получаемые знания в области программирования и отработать их применения на практике.

Список литературы

Приложение 1

```
//
//  main.c
//  Laboratorka
//
//  Created by Kamal Muradov on 27.10.2021.
//

#include <stdio.h>
#include <stdlib.h>

float RandomFloat(int min, int max){
    return ((max - min) * ((double)rand() / RAND_MAX)) + min;
}

int main(){

    int n = 0, min = 0, max = 0, i = 0, indexOfNumber = 1;
    double sum = 0.0;
    double* randomNumber;
    int* randomNumberDecimal;

    printf("Введите кол-во случайных чисел: ");
    scanf("%d", &n);

    printf("Введите минимум диапазона: ");
    scanf("%d", &min);

    printf("Введите максимум диапазона: ");
    scanf("%d", &max);

    printf("\n");

    randomNumber = (double*)malloc(n * sizeof(double));
    randomNumberDecimal = (int*)malloc(n * sizeof(int));

    // создаем случайное число и узнаем десятичную часть числа
    for (i = 0; i < n; i++) {

        randomNumber[i] = RandomFloat(min, max);
        randomNumberDecimal[i] = (randomNumber[i] - (int)randomNumber[i]) *
1000000;

    }
```



```
// проверяем условия для суммирования
for (i = 0; i < n; i++){

    if (randomNumberDecimal[i] == indexOfNumber) {
        sum -= randomNumber[i];
    }
    else {
        sum += randomNumber[i];
    }
    indexOfNumber++;
}
// вывод суммы
printf("Полученная сумма: %f\n", sum);
return 0; // программа успешно завершена
}
```